



Digital  
College

# DESENVOLVEDOR **FULL STACK**

**PROGRAMAÇÃO FRONT-END EM JAVASCRIPT**

UNIDADE 2:

MÓDULO 1:

> **HTML E CSS AVANÇADO** >

## 1. e 2. Flex box

**Fonte:** <https://developer.mozilla.org/> | <https://css-tricks.com/>

Uma nova tecnologia, mas com suporte bastante difundido entre navegadores, o Flexbox, está se tornando apto para uso geral. Flexbox provê ferramentas para criação rápida de layouts complexos e flexíveis, e características que se mostraram historicamente difíceis com CSS. Este artigo explica todos os seus fundamentos.

- **Background**

O Flexbox Layout módulo (Flexible Box) (uma Recomendação Candidata do W3C em outubro de 2017) visa fornecer uma maneira mais eficiente de dispor, alinhar e distribuir o espaço entre os itens em um contêiner, mesmo quando seu tamanho é desconhecido e/ou dinâmico (assim o palavra “flex”).

A principal ideia por trás do layout flexível é dar ao contêiner a capacidade de alterar a largura/altura de seus itens (e ordem) para preencher melhor o espaço disponível (principalmente para acomodar todos os tipos de dispositivos de exibição e tamanhos de tela). Um contêiner flexível expande itens para preencher o espaço livre disponível ou os reduz para evitar estouro.

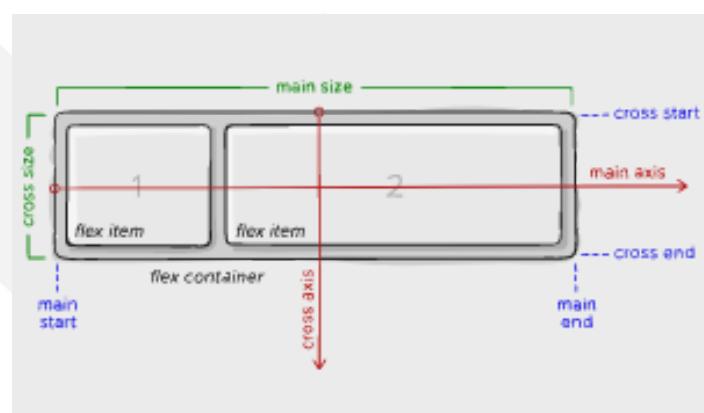
Mais importante ainda, o layout flexbox é independente de direção, em oposição aos layouts regulares (bloco que é baseado na vertical e inline que é baseado na horizontal).

Embora funcionem bem para páginas, eles não têm flexibilidade (sem trocadilhos) para suportar aplicativos grandes ou complexos (especialmente quando se trata de mudança de orientação, redimensionamento, alongamento, redução etc.).

- Noções básicas e terminologia

Como o flexbox é um módulo inteiro e não uma propriedade única, ele envolve muitas coisas, incluindo todo o seu conjunto de propriedades. Alguns deles devem ser definidos no contêiner (elemento pai, conhecido como “contêiner flexível”), enquanto os outros devem ser definidos nos filhos (disse “itens flexíveis”).

Se o layout “regular” é baseado em direções de fluxo em bloco e em linha, o layout flexível é baseado em “direções de fluxo flexível”. Por favor, dê uma olhada nessa figura da especificação, explicando a ideia principal por trás do layout flexível.



Os itens serão dispostos seguindo o main axis (de main-start main-end) ou o eixo cruzado (de cross-start to cross-end).

**main axis** – O eixo principal de um contêiner flexível é o eixo primário ao longo do qual os itens flexíveis são dispostos. Cuidado, não é necessariamente horizontal; depende da flex-direction propriedade.

**main-start** | main-end – Os itens flex são colocados dentro do container começando do main-start e indo para o main-end.

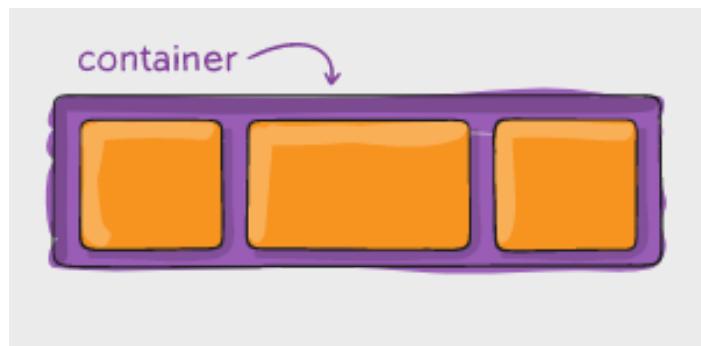
**main size** – a largura ou altura de um item flexível, o que estiver na dimensão principal, é o tamanho principal do item. A propriedade de tamanho principal do item flexível é a propriedade 'largura' ou 'altura', o que estiver na dimensão principal.

**cross axis** – O eixo perpendicular ao eixo principal é chamado de eixo cruzado. Sua direção depende da direção do eixo principal.

**cross-start** | cross-end – As linhas flexíveis são preenchidas com itens e colocadas no contêiner começando no lado de início cruzado do contêiner flexível e indo em direção ao lado de extremidade cruzada.

**cross size** – a largura ou altura de um item flexível, o que estiver na dimensão cruzada, é o tamanho cruzado do item. A propriedade de tamanho cruzado é qualquer 'largura' ou 'altura' que esteja na dimensão cruzada.

- **Flexbox properties**



- **Properties for the Parent (flex container)**

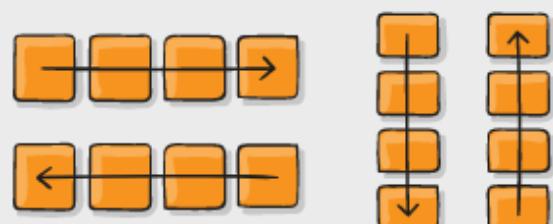
### DISPLAY

Isso define um contêiner flexível; inline ou bloco dependendo do valor fornecido. Ele habilita um contexto flexível para todos os seus filhos diretos.

```
.container {
  display: flex; /* or inline-flex */
}
```

Observe que as colunas CSS não têm efeito em um contêiner flexível.

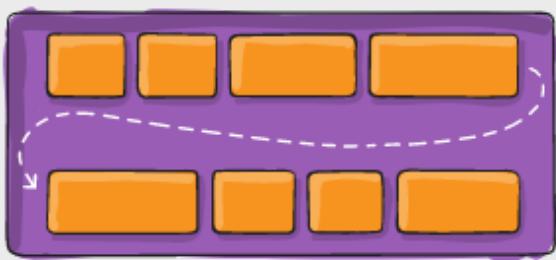
- **Flex-Direction**



Isso estabelece o eixo principal, definindo assim a direção em que os itens flexíveis são colocados no contêiner flexível. Flexbox é (além da embalagem opcional) um conceito de layout de direção única. Pense nos itens flexíveis como principalmente dispostos em linhas horizontais ou colunas verticais.

```
.container {
  flex-direction: row | row-reverse |
  column | column-reverse;
}
```

- **row(padrão):** da esquerda para a direita em ltr; direita para a esquerda em rtl
  - **row-reverse:** da direita para a esquerda em ltr; da esquerda para a direita em rtl
  - **column:** igual, row mas de cima para baixo
  - **column-reverse:** igual, row-reverse mas de baixo para cima
- **flex-wrap**



Por padrão, todos os itens flexíveis tentarão caber em uma linha. Você pode alterar isso e permitir que os itens sejam agrupados conforme necessário com essa propriedade.

```
.container {
  flex-wrap: nowrap | wrap | wrap-
reverse;
}
```

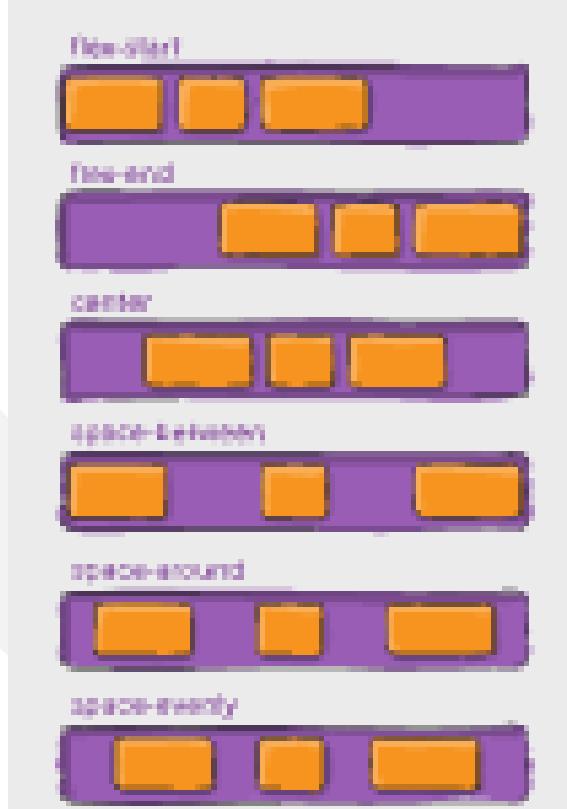
- **nowrap(padrão):** todos os itens flexíveis estarão em uma linha
- **wrap:** os itens flexíveis serão agrupados em várias linhas, de cima para baixo.
- **wrap-reverse:** os itens flexíveis serão agrupados em várias linhas de baixo para cima.

• **flex-flow**

Esta é uma abreviação para as propriedades flex-direction flex-wrap, que juntas definem os eixos principal e cruzado do contêiner flexível. O valor padrão é row nowrap.

```
.container {
  flex-flow: column wrap;
}
```

• **justify-content**



Isso define o alinhamento ao longo do eixo principal. Ele ajuda a distribuir espaço livre extra quando todos os itens flexíveis em uma linha são inflexíveis ou são flexíveis, mas atingiram seu tamanho máximo. Ele também exerce algum controle sobre o alinhamento dos itens quando eles ultrapassam a linha.

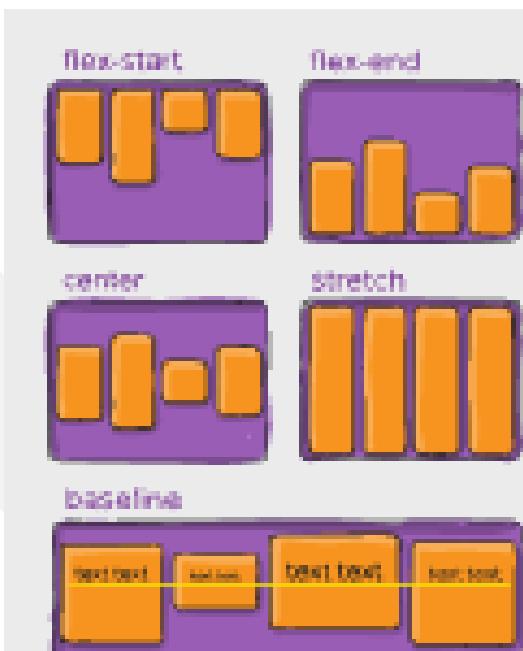
```
.container {
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe;
}
```

- **flex-start(padrão):** os itens são empacotados no início da direção flexível.
- **flex-end:** os itens são empacotados no final da direção flexível.
- **start:** os itens são embalados em direção ao início da writing-mo de direção.
- **end:** os itens são embalados no final da writing-mo de direção.
- **left:** os itens são empacotados em direção à borda esquerda do contêiner, a menos que isso não faça sentido com o flex-direction, então ele se comporta como start.
- **right:** os itens são empacotados na borda direita do contêiner, a menos que isso não faça sentido com o flex-direction, então ele se comporta como start.
- **center:** os itens são centralizados ao longo da linha
- **space-between:** os itens são distribuídos uniformemente na linha; primeiro item está na linha inicial, último item na linha final

- **space-around:** os itens são distribuídos uniformemente na linha com espaço igual ao redor deles. Observe que visualmente os espaços não são iguais, pois todos os itens possuem espaços iguais em ambos os lados. O primeiro item terá uma unidade de espaço contra a borda do contêiner, mas duas unidades de espaço entre o próximo item porque esse próximo item tem seu próprio espaçamento que se aplica.
- **space-evenly:** os itens são distribuídos de forma que o espaçamento entre quaisquer dois itens (e o espaço até as bordas) seja igual.

Observe que o suporte do navegador para esses valores é diferenciado. Por exemplo, space-between nunca recebeu suporte de algumas versões do Edge e o início/fim/esquerda/direita ainda não está no Chrome.

### • align-items



Isso define o comportamento padrão de como os itens flexíveis são dispostos ao longo do eixo cruzado na linha atual. Pense nisso como a justify-content versão para o eixo cruzado

```
.container {
    align-items: stretch | flex-start | flex-end | center | baseline | first baseline | last baseline | start | end | self-start | self-end + ... safe | unsafe;
}
```

- **stretch(padrão):** esticar para encher o contêiner (ainda respeita min-width/max-width)
- **flex-start/ start/ self-start:** os itens são colocados no início do eixo cruzado. A diferença entre eles é sutil, e se trata de respeitar as flex-direction regras ou as writing-mode regras.
- **flex-end/ end/ self-end:** os itens são colocados no final do eixo cruzado. A diferença novamente é sutil e é sobre respeitar flex-direction regras versus writing-mode regras.
- **center:** os itens são centralizados no eixo cruzado
- **baseline:** os itens são alinhados como suas linhas de base se alinham

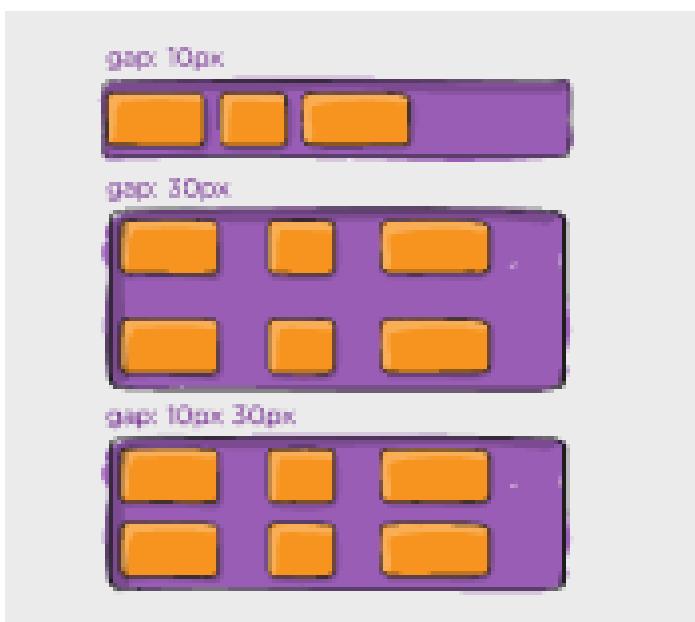
- **align-content**



Isso alinha as linhas de um contêiner flexível quando há espaço extra no eixo cruzado, semelhante a como justify-content alinha itens individuais no eixo principal.

```
.container {
    align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline + ... safe | unsafe;
}
```

- **normal(padrão):** os itens são embalados em sua posição padrão como se nenhum valor fosse definido.
  - **flex-start/ start:** itens embalados até o início do contêiner. O (mais suportado) flex-start honra o flex-direction enquanto start honra a writing-mode direção.
  - **flex-end/ end:** itens embalados até o final do contêiner. O (mais suporte) flex-end honra o flex-direction enquanto o final honra a writing-mode direção.
  - **center:** itens centralizados no contêiner
  - **space-between:** itens distribuídos uniformemente; a primeira linha está no início do contêiner enquanto a última está no final
  - **space-around:** itens distribuídos uniformemente com espaço igual ao redor de cada linha
  - **space-evenly:** os itens são distribuídos uniformemente com espaço igual ao redor deles
  - **stretch:** as linhas se estendem para ocupar o espaço restante
- **gap, row-gap, column-gap**

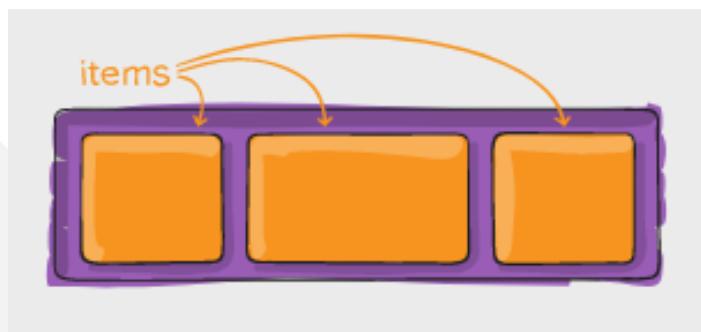


A gap propriedade controla explicitamente o espaço entre os itens flexíveis. Aplica-se este espaçamento apenas entre os itens e não nas bordas externas.

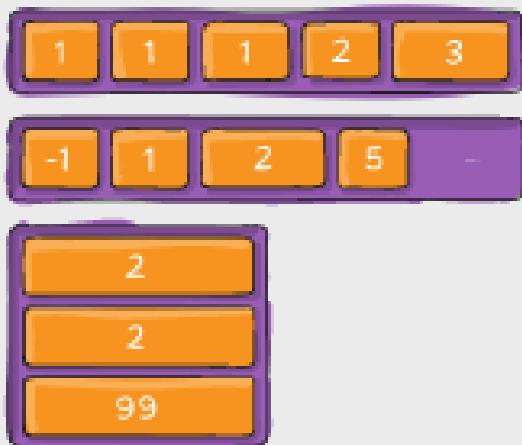
```
.container {
  display: flex;
  ...
  gap: 10px;
  gap: 10px 20px; /* row-gap column
  gap */
  row-gap: 10px;
  column-gap: 20px;
}
```

O comportamento pode ser pensado como uma calha mínima , como se a calha fosse maior de alguma forma (por causa de algo como justify-content: space-between;), a lacuna só terá efeito se esse espaço acabar menor. Não é exclusivo para flexbox, gap funciona também em layout de grade e multi-coluna.

- **Properties for the Children (flex items)**



- **order**

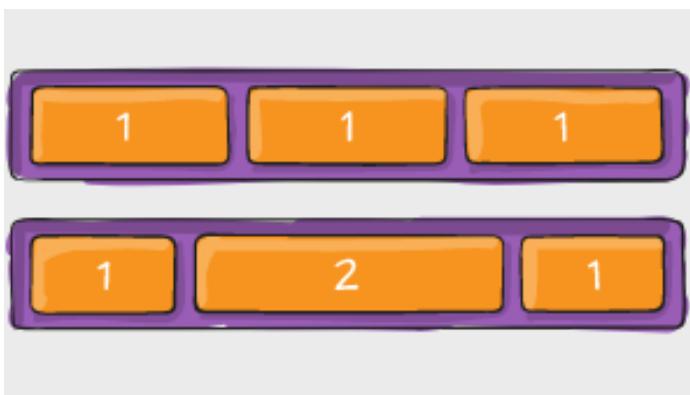


Por padrão, os itens flexíveis são dispostos na ordem de origem. No entanto, a `order` propriedade controla a ordem em que aparecem no contêiner flexível.

```
.item {
  order: 5; /* default is 0 */
}
```

Itens com a mesma `order` reversão para a ordem de origem.

- **flex-grow**



Isso define a capacidade de um item flexível crescer, se necessário. Aceita um valor sem unidade que serve como proporção. Ele determina a quantidade de espaço disponível dentro do contêiner flexível que o item deve ocupar.

Se todos os itens estiverem `flex-grow` definidos como 1, o espaço restante no contêiner será distribuído igualmente para todos os filhos. Se um dos filhos tiver um valor de 2, esse filho ocupará o dobro do espaço de qualquer um dos outros (ou tentará, pelo menos).

```
.item {
  flex-grow: 4; /* default 0 */
}
```

Números negativos são inválidos.

- **flex-shrink**

Isso define a capacidade de um item flexível diminuir, se necessário.

```
.item {
  flex-shrink: 3; /* default 1 */
}
```

Números negativos são inválidos.

- **flex-basis**

Isso define o tamanho padrão de um elemento antes que o espaço restante seja distribuído. Pode ser um comprimento (por exemplo, 20%, 5rem, etc.) ou uma palavra-chave. A palavra-autochave significa “olhar para minha propriedade largura ou altura” (o que foi feito temporariamente pela palavra- `main-size` chave até ser descontinuado).

A palavra- content chave significa “dimensionar com base no conteúdo do item” – essa palavra-chave ainda não é bem suportada, por isso é difícil testar e mais difícil saber o que seus irmãos max-content, min-content fit-content fazem.

```
.item {
  flex-basis: | auto; /* default auto */
}
```

Se definido como 0, o espaço extra em torno do conteúdo não é considerado. Se definido como auto, o espaço extra é distribuído com base em seu flex-grow valor

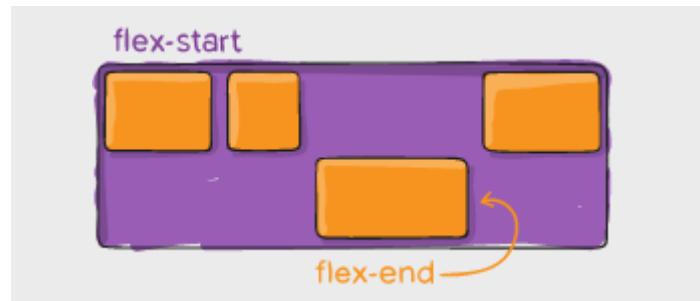
- **flex**

Esta é a abreviação de flex-grow, flex-shrink e flex-basis combinado. O segundo e terceiro parâmetros (flex-shrink, flex-basis) são opcionais. O padrão é 0 1 auto, mas se você defini-lo com um único valor numérico, como flex: 5;, isso muda flex-basis para 0%, então é como configurar flex-grow: 5; flex-shrink: 1; flex-basis: 0%;.

```
.item {
  flex: none | [ <'flex-grow'> <'flex-
shrink'? || <'flex-basis'> ]
}
```

É recomendável que você use essa propriedade abreviada em vez de definir as propriedades individuais. A taquigrafia define os outros valores de forma inteligente.

- **align-self**



Isso permite que o alinhamento padrão (ou o especificado por align-items) seja substituído por itens flexíveis individuais.

Consulte a align-items explicação para entender os valores disponíveis.

```
.item {
  align-self: auto | flex-start |
flex-end | center | baseline | stretch;
}
```

Observe que float, clear e vertical-align não têm efeito em um item flexível.

## 3. e 4. Media query (responsividade)

**Fonte:** [https://developer.mozilla.org/ptBR/docs/Web/CSS/Media\\_Queries/Using\\_media\\_queries](https://developer.mozilla.org/ptBR/docs/Web/CSS/Media_Queries/Using_media_queries),  
<https://www.devmmedia.com.br/utilizando-css-media-queries/27085>

- **Sintaxe**

Uma media query consiste de um media type e pelo menos uma expressão que limita o escopo das folhas de estilo usando media features, tal como largura, altura e cor. Media queries, adicionadas no CSS3, deixam a apresentação do conteúdo adaptado a uma gama específica de dispositivos, não precisando mudar o conteúdo em si.

```
<!-- CSS media query em um
elemento de link -->
<link rel="stylesheet" media="
(max-width: 800px)"
href="example.css" />
<!-- CSS media query dentro de um
stylesheet -->
<style>
@media (max-width: 600px)
{
  .facet_sidebar
  {
    display: none;
  }
}
</style>
```

Quando uma media query é verdadeira, a camada de estilo ou as regras de estilos correspondentes são aplicadas, seguindo o padrão de regras de cascadas. Camadas de estilos com media queries ligadas a tag <link> vão fazer download mesmo se suas media queries retornarem falso (eles não se aplicam, no entanto).

A menos que você use os operadores not ou only, o media type é opcional e o tipo all será implícito.

- **Operadores lógicos**

Você pode compor media queries complexas usando operadores lógicos, incluindo not, and, e only. O operador and é usado para combinar múltiplas media features em uma mesma media query, requerendo que cada sequência de características, retorne verdadeiro na ordem para que a query seja verdadeiro. O operador not é usado para negar uma media query inteira. O operador only é usado para aplicar um estilo apenas se a query inteira for igual, útil para prevenir que navegadores antigos apliquem os estilos selecionados. Se você usar os operadores not ou only, você tem que especificar um tipo de media explícito.

Você também pode combinar múltiplas media queries em uma lista separadas por vírgulas, se qualquer uma das media queries na lista é verdadeira, toda a instrução retorna verdadeira. Isto é equivalente a um operador or.

Para que possamos manter nossas páginas sempre adequadas a cada tipo de visualização, utilizamos os media types, que são listados abaixo de acordo com sua utilização.

- **All:** Para todos os dispositivos.
- **Braille:** Para dispositivos táteis.
- **Embossed:** Para dispositivos que imprimem em braile.
- **Handheld:** Para dispositivos portáteis, geralmente com telas pequenas e banda limitada.
- **Print:** Para impressão em papel.
- **Projection:** Para apresentações como PPS.
- **Screen:** Para monitores ou dispositivos com telas coloridas e resolução adequada.
- **Speech:** Para sintetizadores de voz. As CSS 2 tem uma especificação de CSS chamada Aural, onde podemos formatar a voz dos sintetizadores.
- **Tty:** Para dispositivos que possuem uma grade fixa para exibição de caracteres, tais como: Teletypes, Terminais e também dispositivos portáteis com display limitados.
- **Tv:** Para dispositivos como televisores, ou seja, com baixa resolução, quantidade de cores e scroll limitados.

Agora vamos por o que aprendemos até aqui em prática. Para iniciar vamos começar criando uma folha de estilos CSS com nossas media queries definidas.

```
<style type="text/css">
  body { padding: 15px; background:#eee;
font-family: Arial, Helvetica, sans-serif }
  .caixa {
    border: solid 1px #666;
    padding: 5px 10px;
    margin: 40px;
  }
  .area-visivel span {
    color: #666;
    display: none;
  }
</style>
```

```
/* max-width */
@media screen and (max-width: 600px) {
  .um {
    background: #F9C;
  }
  span.lt600 {
    display: inline-block;
  }
}
```

```
/* min-width */
@media screen and (min-width: 900px)
{
  .dois {
    background: #F90;
  }
  span.gt900 {
    display: inline-block;
  }
}
```

```
/* min-width & max-width */
@media screen and (min-width: 600px)
and (max-width: 900px) {
  .tres {
    background: #9CF;
  }
  span.bt600-900 {
    display: inline-block;
  }
}
```

```
/* max device width */
@media screen and (max-device-width:
480px) {
  .iphone {
    background: #ccc;
  }
}
</style>
```

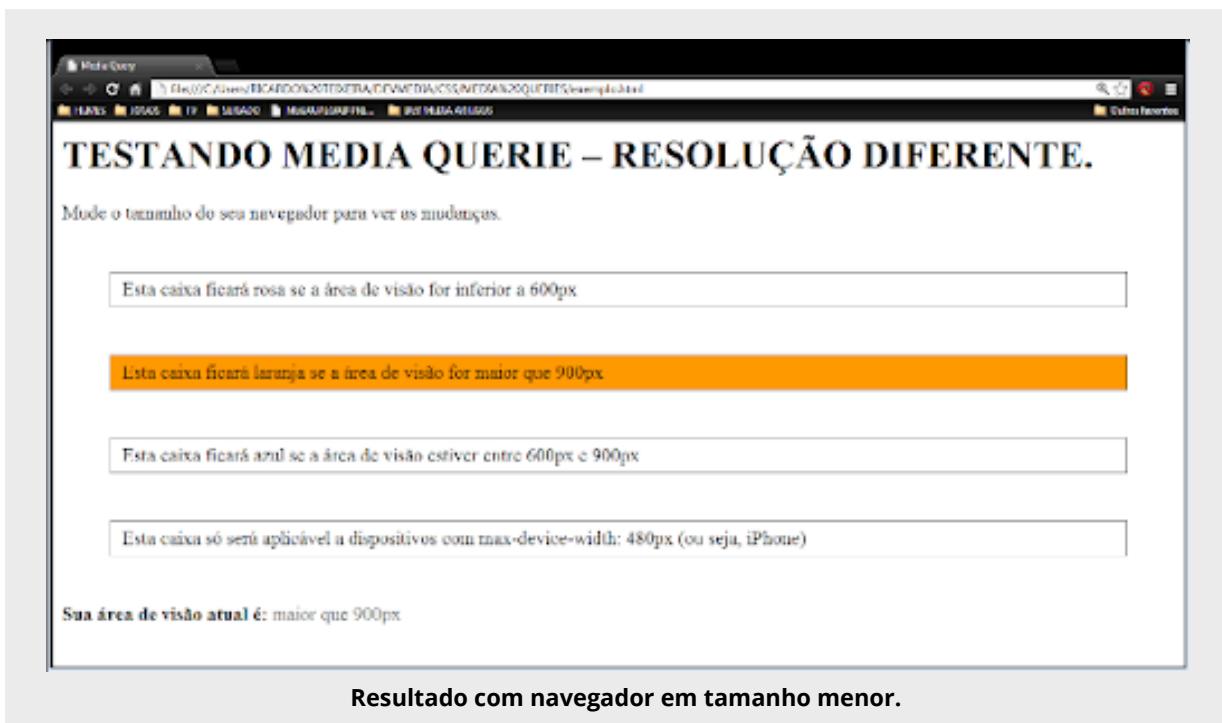
Agora que já criamos nossa folha de estilos CSS, vamos salvar como estilo.css. Vamos agora criar a página HTML e carregar a nossa folha de estilos que acabamos de criar.

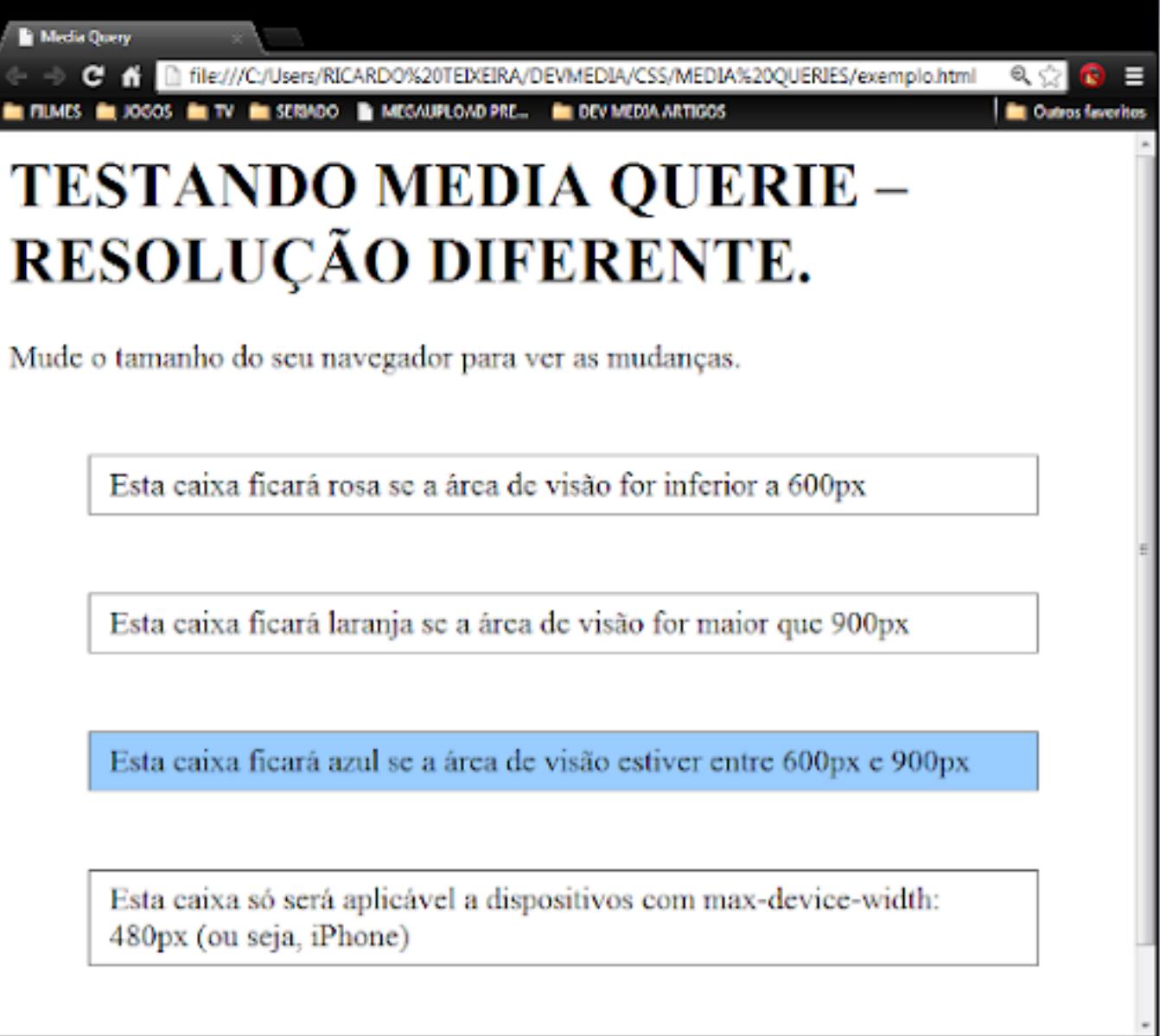
```

<head>
<title>Media Query</title>
<link rel="stylesheet" href="estilo.css" /> <!--Folha de estilo criada
anteriormente carregada -->
</head>
<body>
<h1>TESTANDO MEDIA QUERIE - RESOLUÇÃO DIFERENTE. </h1>
<p>Mude o tamanho do seu navegador para ver as mudanças.</p>
<div class="caixa um">Esta caixa ficará rosa se a área de visão for inferior a
600px</div> <div class="caixa dois">Esta caixa ficará laranja se a área de
visão for maior que 900px</div>
<div class="caixa tres">Esta caixa ficará azul se a área de visão estiver
entre 600px e 900px</div>
<div class="wrapper iphone">Esta caixa só será aplicável a dispositivos com
max-device-width: 480px (ou seja, iPhone)</div> <p class="area-visivel">
<strong>Sua área de visão atual é: </strong>
<span class="lt600">menor que 600px</span>
<span class="bt600-900"> entre 600 - 900px</span>
<span class="gt900">maior que 900px</span>
</p>
</body>
</html>

```

Salvamos esse arquivo HTML como index.html e abrimos com um navegador de nossa preferência para testarmos. O resultado é o seguinte:





A screenshot of a Microsoft Edge browser window titled "Media Query". The address bar shows the local file path: "file:///C/Users/RICARDO%20TEIXEIRA/DEVMEDIA/CSS/MEDIA%20QUERIES/exemplo.html". The page content is a large title:

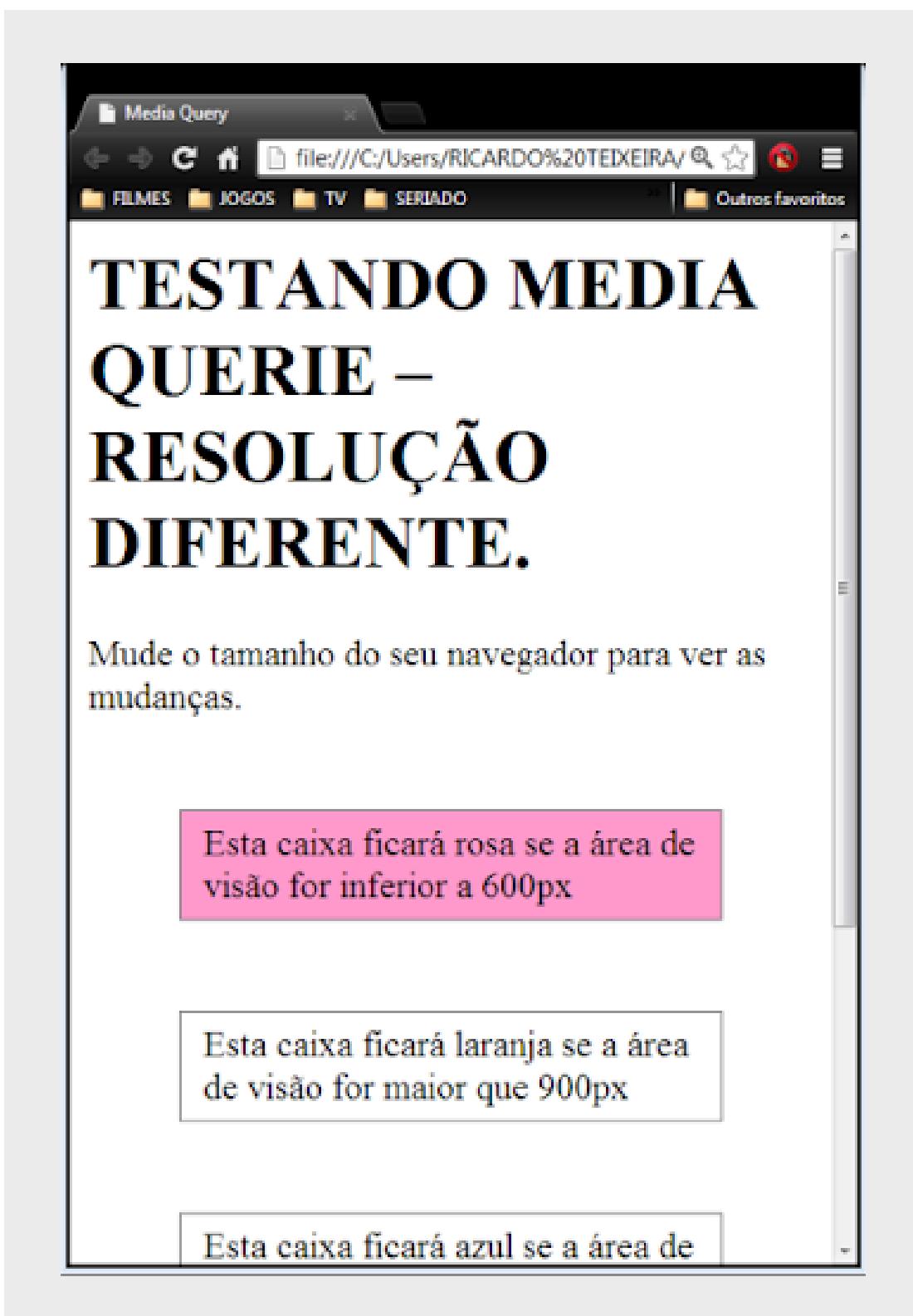
# TESTANDO MEDIA QUERIE – RESOLUÇÃO DIFERENTE.

Below the title, there is a instruction: "Mude o tamanho do seu navegador para ver as mudanças." (Change the size of your browser to see the changes.)

The page contains four boxes, each with a different background color and a descriptive text message:

- A white box with a black border: "Esta caixa ficará rosa se a área de visão for inferior a 600px"
- A white box with a black border: "Esta caixa ficará laranja se a área de visão for maior que 900px"
- A light blue box with a black border: "Esta caixa ficará azul se a área de visão estiver entre 600px e 900px"
- A white box with a black border: "Esta caixa só será aplicável a dispositivos com max-device-width: 480px (ou seja, iPhone)"

At the bottom of the page, there is a final note: "E para finalizar, o navegador com uma resolução abaixo de 600px."



Vemos que a página, com a simples aplicação de media queries, se comportou de formas diferentes, de acordo com a resolução da tela.

## 5. e 6. Abordagem de layouts (aplicando conceitos de flex box)

O professor irá propor uma atividade para utilizar os conhecimentos adquiridos

## 7. e 8. Abordagem de layouts (aplicando conceitos de responsividade)

O professor irá propor uma atividade para utilizar os conhecimentos adquiridos

## 9. 10. e 11. Bootstrap

### Fonte:

<https://getbootstrap.com.br/docs/4.1/layout/overview/>

O framework mais conhecido do mundo para criar sites responsivos e mobile, comece com o BootstrapCDN e nosso template inicial.

### CSS

Copie e cole o arquivo de estilo <link> dentro da sua <head> antes de todos os outros arquivos de estilo para carregar nosso CSS.

```
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXF
      oaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
      crossorigin="anonymous">
```

### JS

Muitos dos nossos componentes precisam de JavaScript para funcionar. Mais especificamente, eles precisam do jQuery, Popper.js e do nossos próprios plugins JavaScript. Coloque os seguintes <script>s perto do final da sua página, logo antes do fechamento da tag </body>. Query tem que vir antes, depois o Popper.js e só depois nossos plugins JavaScript.

Nós usamos a build slim do jQuery, mas a versão completa também é suportada.

```
<script
      src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
      integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRV
      zpbzo5smXKp4YfRvH+8abTE1Pi6jizo
      " crossorigin="anonymous">
</script>
<script
      src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
      integrity="sha384-zMP7rVo3mIykV+2+9J3UJ46jBk0WLau
      dn689aCwoqbBJiSnjAK/l8WvCWPIPm49
      " crossorigin="anonymous">
</script>
<script
      src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IM
      qE241rYiqJxyMiZ6OW/JmZQ5stwEULTy
      " crossorigin="anonymous">
</script>
```

### Template inicial

Tenha certeza de configurar suas páginas com padrões recentes de desenvolvimento e design. Ou seja, utilizando HTML5 doctype e a meta tag viewport para proporcionar o funcionamento responsivo adequado. Feito isto, suas páginas devem ficar assim

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <!-- Meta tags Obrigatórias -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.
min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">

      <title>Olá, mundo!</title>
    </head>
    <body>
      <h1>Olá, mundo!</h1>

      <!-- JavaScript (Opcional) -->
      <!-- jQuery primeiro, depois Popper.js, depois Bootstrap JS -->
      <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965Dz00rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo"
crossorigin="anonymous"></script>
      <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.
min.js" integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jbk0WLauAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>
      <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.mi
n.js" integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
    </body>
  </html>
```

- **BootstrapCDN**

Evite o download, usando a BootstrapCDN para ter uma versão em cache dos CSS e JS compilados, em seu projeto.

```
<link rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXWEOngsV7Zt27NXFOaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
      crossorigin="anonymous">
<script
      src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
      crossorigin="anonymous"></script>
```

Se você está usando nosso JavaScript compilado, não esqueça de incluir as versões em CDN do jQuery e Popper.js, antes dele.

```
<script
      src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
      integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzp
      bzo5smXKp4YfRvH+8abTE1Pi6jizo"
      crossorigin="anonymous"></script>
<script
      src="https://cdnjs.cloudflare.com/
      ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLauAdn
      689aCwoqbBJiSnjAK/l8WvCWPIPm49"
      crossorigin="anonymous"></script>
```

- **Visão Geral**

Componentes e opções para o layout do seu projeto Bootstrap, incluindo wrapping containers, um poderoso sistema de grid, um objeto de mídia flexível e classes responsivas.

- **Containers**

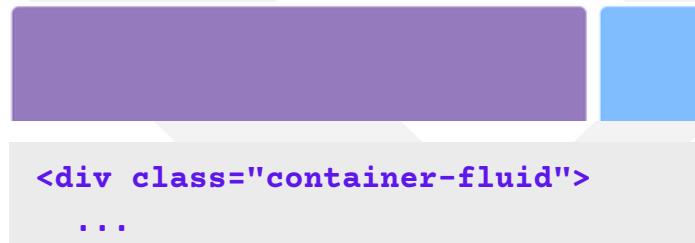
Containers são os elementos de layout mais básico do Bootstrap e são necessários quando usamos o sistema de grid padrão. Escolha entre um container responsável de largura fixa (ou seja, com alterações de max-width em cada ponto de interrupção) ou por um responsável de largura fluida (ou seja, 100% de largura o tempo todo).

Embora os containers possam ser aninhados, a maioria dos layouts não exige um container aninhado.



```
<div class="container">
  <!-- Conteúdo aqui -->
</div>
```

Use .container-fluid para um container com de largura total, abrangendo toda a largura da sua área de visualização.



```
<div class="container-fluid">
  ...
</div>
```

- **Breakpoints responsivos**

Como o bootstrap é desenvolvido para ser “mobile first” usamos várias media queries para criar sensíveis breakpoints para nossos layouts e interfaces. Esses pontos são baseados em larguras de área de visualização mínima e nos permitem dimensionar conforme muda o tamanho da área de visualização.

Bootstrap usa principalmente os seguintes intervalos de media queries, ou breakpoints, em nossos arquivos de origem do SASS para os layouts, sistemas de grid e componentes.

**// Dispositivos extra small (telefones em modo retrato, com menos de 576px)**

**// Sem media query para `xs`, já que este é o padrão, no Bootstrap.**

**// Dispositivos small (telefones em modo paisagem, com 576px ou mais)**

**@media (min-width: 576px) { ... }**

**// Dispositivos médios (tablets com 768px ou mais)**

**@media (min-width: 768px) { ... }**

**// Dispositivos large (desktops com 992px ou mais)**

**@media (min-width: 992px) { ... }**

**// Dispositivos extra large (desktops grandes com 1200px ou mais)**

**@media (min-width: 1200px) { ... }**

- **Alguns Componentes**

### Alertas

Apresente mensagens contextuais para ações típicas dos usuários, usando este punhado flexível de mensagens de alerta.

### Exemplos

Alertas estão disponíveis para qualquer tamanho de texto, assim como um botão de dispersão opcional. Para uma estilização adequada, use uma das oito requeridas classes contextuais (ex: .alert-success). Para dispersão inline, use o plugin jQuery alerts.

Um simples alerta primary. Olha só!

Um simples alerta secondary. Olha só!

Um simples alerta success. Olha só!

Um simples alerta danger. Olha só!

Um simples alerta warning. Olha só!

Um simples alerta info. Olha só!

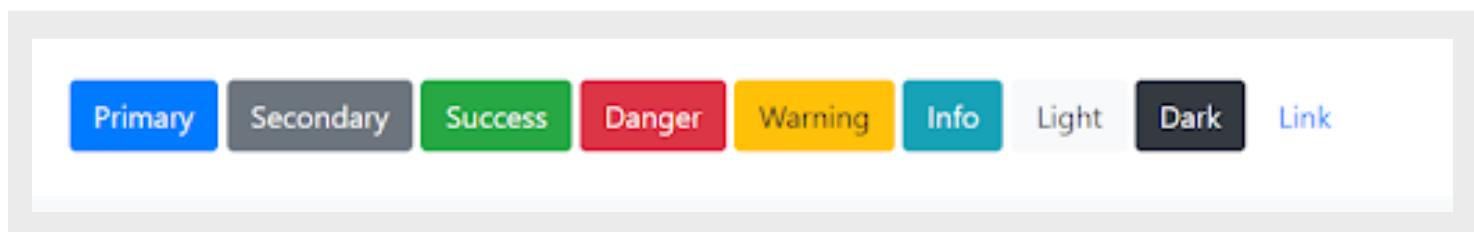
```
<div class="alert alert-primary" role="alert">  
  Um simples alerta primary. Olha só!  
</div>  
<div class="alert alert-secondary" role="alert">  
  Um simples alerta secondary. Olha só!  
</div>  
<div class="alert alert-success" role="alert">  
  Um simples alerta success. Olha só!  
</div>  
<div class="alert alert-danger" role="alert">  
  Um simples alerta danger. Olha só!  
</div>  
<div class="alert alert-warning" role="alert">  
  Um simples alerta warning. Olha só!  
</div>  
<div class="alert alert-info" role="alert">  
  Um simples alerta info. Olha só!  
</div>  
<div class="alert alert-light" role="alert">  
  Um simples alerta light. Olha só!  
</div>  
<div class="alert alert-dark" role="alert">  
  Um simples alerta dark. Olha só!  
</div>
```

## Botões

Use os botões Bootstrap personalizados para ações em formulários, diálogos e outras coisas. Lembrando que esse componente tem suporte a escolha de tamanhos, estados e muito mais.

## Exemplos

Bootstrap possui vários estilos de botões pré-definidos, cada um com seu propósito semântico e outros recursos extras para mais controle.



```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>
<button type="button" class="btn btn-link">Link</button>
```

## Formulários

Exemplos, instruções para estilizar campos de formulários, opções de layout e componentes personalizados para criar uma grande variedade de formulários.

## Visão geral

Campos de formulários Bootstrap extendem os estilos herdados do Reboot, graças a classes. Use estas classes para conseguir uma exibição personalizada e, portanto, uma renderização nos browsers e dispositivos, mais consistentes.

Se assegure de usar um atributo type adequado em todos inputs (email em campos de email, por exemplo) para conseguir se aproveitar dos novos inputs, como seletores de números, verificadores de e-mails, etc.

Aqui está um rápido exemplo para demonstrar os estilos de formulário Bootstrap. Continue lendo a documentação para descobrir sobre classes obrigatórias, layout de formulários e muito mais.

Endereço de email

 Seu emailNunca vamos compartilhar seu email, com ninguém.

Senha

 Senha Clique em mimEnviar

```
<form>
  <div class="form-group">
    <label for="exampleInputEmail1">Endereço de email</label>
    <input type="email" class="form-control" id="exampleInputEmail1" aria-
describedby="emailHelp" placeholder="Seu email">
    <small id="emailHelp" class="form-text text-muted">Nunca vamos compartilhar
seu email, com ninguém.</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Senha</label>
    <input type="password" class="form-control" id="exampleInputPassword1"
placeholder="Senha">
  </div>
  <div class="form-group form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Clique em mim</label>
  </div>
  <button type="submit" class="btn btn-primary">Enviar</button>
</form>
```

## • Campos de formulário

Campos de formulário textuais como `<input>`, `<select>` e `<textarea>` são estilizados com a classe `.form-control`. Ela possui estilos para aparência, estado de foco e muito mais. Tenha certeza de explorar nossos formulários personalizados para estilos de `<select>`.

Endereço de email

Select de exemplo

1

Exemplo de select múltiplo

1

2

3

4

Exemplo de textarea

```
<form>
  <div class="form-group">
    <label for="exampleFormControlInput1">Endereço de email</label>
    <input type="email" class="form-control" id="exampleFormControlInput1"
placeholder="nome@exemplo.com">
  </div>
  <div class="form-group">
    <label for="exampleFormControlSelect1">Select de exemplo</label>
    <select class="form-control" id="exampleFormControlSelect1">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
      <option>5</option>
    </select>
  </div>
  <div class="form-group">
    <label for="exampleFormControlSelect2">Exemplo de select múltiplo</label>
    <select multiple class="form-control" id="exampleFormControlSelect2">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
      <option>5</option>
    </select>
  </div>
  <div class="form-group">
    <label for="exampleFormControlTextarea1">Exemplo de textarea</label>
    <textarea class="form-control" id="exampleFormControlTextarea1" rows="3">
    </textarea>
  </div>
</form>
```

## Modal

Use o plugin JavaScript modal para criar diálogos em seu site para notificações e outros tipos de conteúdos customizados.

## Como funciona

Antes de começar a usar o modal Bootstrap, se assegure de ler o seguinte, já que nossas opções de menu mudaram recentemente.

- Modals são feitos com HTML, CSS e JavaScript;
- Eles são posicionados acima de todas as coisas do documento e removem rolagem do `<body>` para que o conteúdo do modal role.
- Clicar na tela atrás do modal vai fazer com que ele seja fechado, automaticamente;
- Bootstrap só suporta uma janela modal, por vez;
- Modals aninhados não são suportados, já que acreditamos proporcionarem pobres experiências ou usuário.
- Modals usam **position: fixed**, o que pode ser um pouco chatinha, no que diz a respeito de sua renderização;
- Sempre que possível, coloque o HTML do seu modal em uma posição prioritária para evitar potenciais interferências de outros elementos;
- Você vai passar por problemas, quando aninhar um **.modal** com outro elemento fixo.
- Mais uma vez, devido a **position: fixed**, existem alguns problemas sobre o uso de modais em dispositivos móveis;
- Veja nossa documentação de suporte a browsers, para mais detalhes.
- Dado como HTML5 define suas semânticas, o atributo HTML autofocus não tem efeito em modals Bootstrap.
- Para alcançar o mesmo efeito, use algum tipo de JavaScript customizado:

```
$('#meuModal').on('shown.bs.modal', function () {  
    $('#meuInput').trigger('focus')  
})
```

Continue lendo, para demonstrações e instruções de uso

## Exemplos

Componentes modal

Abaixo, temos um exemplo de modal static (**position e display sobrescritos**). Pode-se ver o cabeçalho, corpo (precisa de **padding**) e footer (opcional) do modal. Nós só pedimos que você crie cabeçalhos modal com recursos de dispensa, sempre que possível, ou dê outra opção de dispensa ao usuário.

```
<div class="modal" tabindex="-1" role="dialog">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Título do modal</h5>
        <button type="button" class="close" data-dismiss="modal" aria-
label="Fechar">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <p>Texto do corpo do modal, é aqui.</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-
dismiss="modal">Fechar</button>
        <button type="button" class="btn btn-primary">Salvar mudanças</button>
      </div>
    </div>
  </div>
```

## Navs

Documentação e exemplos de como usar os componentes de navegação Bootstrap.

### Nav: a base de tudo

A navegação disponível no Bootstrap compartilha várias marcações e estilos, desde a classe base `.nav` até os estados ativo e desativo. Troque de classes modificadoras para mudar de estilo.

O componente `.nav` é feito com flexbox e provê uma forte fundação para construir todo tipo de componente de navegação. Ela possui sobrescrição de alguns estilos para trabalhar com listas, padding em links para criar áreas de clique maiores e estilo desativado básico.

O componente base `.nav` não inclui qualquer estado `.active`. Os exemplos seguintes possuem a classe, principalmente, para demonstrar que esta classe não aciona nenhum estilo, em específico.

As classes são usadas por toda parte, então, sua marcação consegue ser muito flexível. Use um `<ul>` como acima ou faça seu próprio, com um elemento `<nav>`. Devido o `.nav` usar `display: flex`, os links navs se comportam da mesma forma que os itens nav, mas sem marcação HTML a mais.

Ativo   Link   Link   Desativado

```
<nav class="nav">
  <a class="nav-link active"
  href="#">Ativo</a>
  <a class="nav-link"
  href="#">Link</a>
  <a class="nav-link"
  href="#">Link</a>
  <a class="nav-link disabled"
  href="#">Desativado</a>
</nav>
```

## Texto

Documentação e exemplos de utilitários de texto comuns para controle de alinhamento, quebra de linha, espessura, etc.

### Alinhamento de texto

Facilmente, alinhe texto de componentes com as classes de alinhamento.

Ambitioni dedisse scripsisse iudicaretur. Cras mattis iudicium purus sit amet fermentum. Donec sed odio operae, eu vulputate felis rhoncus. Praeterea iter est quasdam res quas ex communi. At nos hinc posthac, sitientis piros Afros. Petierunt uti sibi concilium totius Galliae in diem certam indicere. Cras mattis iudicium purus sit amet fermentum.

```
<p class="text-justify">Ambitioni
dedisse scripsisse iudicaretur.
Cras mattis iudicium purus sit amet
fermentum. Donec sed odio operae,
eu vulputate felis rhoncus.
Praeterea iter est quasdam res quas
ex communi. At nos hinc posthac,
sitientis piros Afros. Petierunt
uti sibi concilium totius Galliae
in diem certam indicere. Cras
mattis iudicium purus sit amet
fermentum.</p>
```

Para alinhamento à esquerda, direita e centralizado, existem classes responsivas que usam os mesmos breakpoints que o sistema de grid.

Alinhado à esquerda, em todos tamanhos de viewport.

Centralizado, em todos tamanhos de viewport.

Alinhado à direita, em todos tamanhos de viewport.

Alinhado à esquerda, em viewports com tamanho SM (pequenas) ou maior.

Alinhado à esquerda, em viewports com tamanho MD (médias) ou maior.

Alinhado à esquerda, em viewports com tamanho LG (grandes) ou maior.

Alinhado à esquerda, em viewports com tamanho XL (extra-grandess) ou maior.

```
<p class="text-left">Alinhado à
esquerda, em todos tamanhos de
viewport.</p>
<p class="text-center">Centralizado,
em todos tamanhos de viewport.</p>
<p class="text-right">Alinhado à
direita, em todos tamanhos de
viewport.</p>
<p class="text-sm-left">Alinhado à
esquerda, em viewports com tamanho SM
(pequenas) ou maior.</p>
<p class="text-md-left">Alinhado à
esquerda, em viewports com tamanho MD
(médias) ou maior.</p>
<p class="text-lg-left">Alinhado à
esquerda, em viewports com tamanho LG
(grandes) ou maior.</p>
<p class="text-xl-left">Alinhado à
esquerda, em viewports com tamanho XL
(extra-grandess) ou maior.</p>
```

```
<nav class="nav">
  <a class="nav-link active"
  href="#">Ativo</a>
  <a class="nav-link"
  href="#">Link</a>
  <a class="nav-link"
  href="#">Link</a>
  <a class="nav-link disabled"
  href="#">Desativado</a>
</nav>
```

### Quebra e transbordamento de texto

Evite que o texto quebre, usando a classe **.text-nowrap**.

Este texto está transbordando o elemento pai.

```
<div class="text-nowrap bd-
highlight" style="width: 8rem;">
  Este texto está transbordando o
  elemento pai.
</div>
```

Para conteúdos longos demais, você pode usar a classe **.text-truncate** para reduzir o texto com reticências. Requer o uso de **display: inline-block** ou **display: block**.

Praeterea iter est quasdam res quas ex communi.

Praeterea iter est quasdam res quas ex communi.

```
<!-- Elemento com `display: block;` -->
<div class="row">
  <div class="col-2 text-
  truncate">
    Praeterea iter est quasdam res
    quas ex communi.
  </div>
</div>

<!-- Elemento com `display:
inline;` -->
<span class="d-inline-block text-
truncate" style="max-width:
150px;">
  Praeterea iter est quasdam res
  quas ex communi.
</span>
```

### Text transform

Transforme textos com as seguintes classes:

texto em letras minúsculas.

TEXTO EM LETRAS MAIÚSCULAS.

Texto Capitalizado.

```
<p class="text-lowercase">TEXTO EM
LETRAS MINÚSCULAS.</p>
<p class="text-uppercase">Texto em
letras maiúsculas.</p>
<p class="text-capitalize">texto
capitalizado.</p>
```

Perceba que **.text-capitalize** apenas altera a primeira letra de cada palavra.

Espessura e utilização de fontes

Rapidamente, modifique a espessura do texto ou italicize-o.

Texto grosso.

Texto de espessura comum.

Texto de espessura leve.

Texto itálico.

```
<p class="font-weight-bold">Texto  
grosso.</p>  
<p class="font-weight-normal">Texto  
de espessura comum.</p>  
<p class="font-weight-light">Texto  
de espessura leve.</p>  
<p class="font-italic">Texto  
itálico.</p>
```

## Monospace

Use nossa fonte monospace, usando a classe **.text-monospace**.

Este é um texto com fonte monospace.

```
<p class="text-monospace">Este é um  
texto com fonte monospace.</p>
```

para conhecer mais sobre demais componentes acesse:

<https://getbootstrap.com.br/docs/4.1/components/>



# Digital College

ENSINO DE HABILIDADES DIGITAIS

**[digitalcollege.com.br](http://digitalcollege.com.br)**